2. Streaming Technology

2.1 Audio Processing

In telematic performances, audio, video and, in some cases, other types of data, are sent from one location to the other - and vice versa. In this chapter, we are going to introduce some basic principles for streaming such data. This might help users make the right choices and may also help with trouble shooting issues.

The chapter focuses specifically on audio streaming, but keep in mind, many aspects are very similar when it comes to streaming video data.

Discretization

When we transmit sound over the Internet, it experiences a few transformations along its way. Sound waves that propagate as small and fast fluctuations of air pressure are picked up by a microphone and translated into a continuous electrical signal. For processing this signal further with a computer, the signal is digitized with a analog-digital-converter. This reduces the amount of information of an analogue or continuous signal which, theoretically, is infinite. This reduction of information happens in two dimensions.

Samplerate

First, the signal is transformed from continuous time to discrete time by measuring values at regular time intervals. We discard all the information between those points in time. The rate at which we take measurements is called sample rate and is measured in Hertz. Common sampling rates are 44.1 or 48 kHz.

Bitrate

Second, each measurements is quantized by assigning it a value from a finite set of possible values. The amount of possible values is measured in bits. This bitrate influences sound quality and defines the dynamic range reproducible by the digitized signal. Common bitrates are 16, 24 or 32 bit. 16 bit means that our measurement is assigned one of 65 thousand possible values (2^16 = 65536).

Samplerate and bitrate are critical parameters when streaming audio since they directly influence the amount of data that is transmitted in a given time frame.

Blocksize

The current form of our signal is still unsuited for transmission over IP networks. The internet protocol is the fundament of today's Internet and allows computers to exchange data as packets. This means we need to discretize our digital signal further by casting it into packets. A common practice is to create blocks of audio samples with a given size, its 'blocksize'. These blocks can then be sent as packets through computer networks. It's common to put a power-of-two number audio samples into a block, for instance 64 or 128.

The blocksize - among other factors - directly influences latency, since we need to collect enough audio samples before we can send a block as a packet. The higher the blocksize, the higher the latency induced by packetizing.

Compression

Sometimes, the amount of data transmitted is further reduced by compressing it with what is called an 'audio codec'. Many codecs use psycho-acoustical models of our acoustic perception to skip less important parts of the audio signal, while maintaining the more important parts. Most audio transmission systems that employ this kind of compression use the 'OPUS' codec, since it has a good compression ratio for a wide range of targeted audio qualities and is optimized for low latency.

2.2 Audio Transmission

Transmission Protocol

A variety of protocols are used for network based audio transmission, for instance JackTrip or Audio-over-OSC. A protocol defines how data is packetized and labelled with metadata, like its samplerate or its number of channels. All programs concerned with latency use a 'User Datagram Protocol' or 'UDP', as their underlying protocol. The UDP sends packets of data to a destination defined by IP address and port number. Unlike its sibling protocol TCP, which is widely used, it gives absolutely no guarantees about the correct order of transmission. So, in congested networks, UDP packets can be dropped. Because it lacks mechanisms to re-send dropped packets, packets either arrive quickly – or not at all. This behavior is suitable for low-latency protocols, since late packets would miss their targeted time window anyway.

Transmission topology

We can distinguish between two transmission modes. Proxy mode and peer-to-peer mode.

In proxy mode, clients send their data to a central server acting as a proxy, which relays incoming data to the peers (star topology).

In peer-to-peer mode, clients send their data directly to their peers (mesh topology). While a peer-to-peer connection usually means lower latency - due to shorter distance, it's not always feasible to establish a connection between clients, especially when they are located behind a firewall (which they usually are). A technique called UDP hole punching is employed to establish connections through firewalls. However, this technique still requires a central server for the clients to exchange connection information. With certain firewall types, peer-to-peer connections cannot be established at all.

Establishing a connection through a central proxy server works more reliably. The number of clients a proxy server can serve is limited by its own network bandwidth and CPU power. Depending on client locations, a proxy connection may add latency due to longer paths.

2.3 Audio Playback and Quality

Playback

Due to the irregularity of the time they need to travel, individual packets are buffered on the receiving end before they are played back. This time interval difference is called jitter. The higher the jitter, the larger the buffer required for distortion-free playback. If the packet is encoded with an audio codec, the data needs to be uncompressed, before each block of samples is written to the buffer. For the playback, samples are read from the buffer and sent to a digital analog converter that converts the stream of numbers into a continuous electrical signal.

Transmission Quality

Rarely are networked low latency audio transmissions error-free. Since IP networks do not guarantee a high quality of service and work on a best-effort basis, we assume that some packets are delivered late or not at all. The acceptable rate of lost packets depends on the situation. Even for networked concerts, some packet losses are acceptable. Also, the type of transmitted sound matters. In periods of silence, dropped packets might not be noticed at all. Glitches are less audible in noisy sounds than in constant-pitch sounds. Often, the rate of dropped packets can be reduced by increasing the receive buffer. Also, the packet-size has an influence on transmission quality. While UDP supports packets up to 64kB, packets that are larger than 1500 bytes are often fragmented during transmission. Usually, we want to avoid packet fragmentation, since a lost fragment leads to the loss of the whole packet. We also want to avoid sending packets that are too small, because the bandwidth overhead is larger with smaller packets. If we get many dropped packets due to network bandwidth saturation, we need to reduce the amount of the transmitted data per time frame. We can do so by lowering the number of channels, lowering the sample-rate or bit-rate or by compressing the audio with a suitable codec like OPUS.

For transmitting one uncompressed channel at a sample-rate of 44.1kHz and a bitrate of 16bits, we require a bandwidth of approximately 100kbytes per second. With this number in mind, we can roughly calculate the required up- or download bandwidth for sending or receiving a certain number of channels. It is helpful to note that some bandwidth headroom might be needed, since the likelihood of dropped packets increases when nearing bandwidth saturation.